Projet Manifold Learning
Algorithme de réduction de dimension

Lia Furtado and Hugo Vinson

Université Lumière Lyon 2, France

lia.furtado@univ-lyon2.fr, hugo.vinson@univ-lyon2.fr

1 Objectifs

La réduciton de dimensions est un enjeu majeur dans l'apprentissage automatique. Plusieurs études de cas l'ont démontré : lorsque l'on souhaite inférer un model à partir d'un jeu de données, bien souvent la structure de ces dernières est noyée dans une dimension plus grande. Nous parlons ainsi de dimension intrinsèque qui est la dimension réelle des jeux de données et de dimension extrinsèque : la dimension du jeu de donnée. Travailler avec des données en grande dimensions est un problème dans beaucoup d'études car la plupart des algorithmes d'apprentissage perdent en efficacité en raison des propriétés particulières des données en grande dimensions. Pour toutes ces raisons des algorithmes sont nés afin de réduire la dimension des jeux de données. Leur tâche consiste à trouver les dimensions intrinsèques en espérant que cet espace soit de suffisamment faible dimension pour que cela pallie le fléau des dimensions. L'objectif de notre travail est d'étudier ces différents algorithmes de réduction de dimensions, nous les testerons sur des jeux de données artificielles et réels. Les jeux de données artificiels seront simulés suivant différentes techniques. Pour chaque méthodes l'objectif sera de les calibrer au mieux afin d'avoir la meilleure représentation possible de nos données. Enfin nous comparerons les performances des différents algorithmes sur ces jeux de données en proposant un critère de comparaison.

Les principaux objectifs de ce travail sont :

1. Simuler trois jeux de données différents et en choisir un réel.
2. Utiliser différents algorithme de réduction de dimension et choisir pour chaque datasets le plus pertinent.
3. Comparer ces méthodes en utilisant un critère qui prend en compte le temps de calcul, la fiabilité, la continuité et l'esthétique de la nouvelle représentation.

2 Matériel et Méthode

Dans cette partie nous présenterons les différentes méthodes utilisées ainsi que le moyen de les implémenter. Toutes les méthodes implémentées ci-dessous ont été programmé dans le language R (version 4.1.2) en utilisant l'IDE RStudio (version 2022.02.0).

Le lien github avec tout le code et les expériences est: [1]

2.1 Techniques de réduction de la dimensionnalité

Nous avons décidé d'utiliser trois méthodes de réduction que nous présentons ci-dessous.

2.1.1 Multidimensional Scaling

Le premier algorithme choisi est Multidimensional Scaling (MDS) il s'agit d'une méthode linéaire classique. Il s'agit d'une méthode non supervisée qui vise à préserver la distance par paire de l'espace extrinsèque. L'objectif principal est de trouver les coordonnées en basse dimension de chaque point de manière à ce que la distance entre les points d_{ij} soit proche de la distance des points d'_{ij} en haute dimension. Le problème peut donc être vu comme le problème d'optimisation suivant :

$$min(\sum_{i=1}^{m}\sum_{j=1}^{m}(d_{ij} - d'_{ij})^2) \tag{1}$$

Cette approche présente différents points faibles : étant basée sur la notion de distance elle se retrouve moins efficaces si l'espace extrinsèque est de très grande dimension. Ensuite, cette méthode est sensible aux outliers. Enfin, elle est limitée quand les transformations pour passer de la structure en dimension extrinsèque à la structure en dimension intrinsèque nécessite des transformations non linéaire. Nous avons utilisé le package mds implémenté dans la fonction *cmdscale* de la bibliothèque stats en R. [2]

2.1.2 Isomap

La deuxième technique est Isomap, une méthode non linéaire, elle utilise se base sur la distance géodésique pour trouver la représentation en dimension inférieure des données . La distance géodésique est une distance calculée en utilisant le graphe des voisinages d'un point. Chaque point est un noeud du graphe et un arc existe entre ce

[1] https://github.com/hugomdm/manifold_project

[2] https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/cmdscale

point et ces voisins, la distance de voisinages est un hyperparamètre. Enfin la distance entre deux point est calculée par la notion de plus court chemin (de la théorie des graphes). Cette approche présente deux principaux inconvénients, le nombre de voisins doit être paramétrer cela peut nécessiter un ajustement de la méthode. Ainsi suivant le paramètre choisi cela peut posé un problème si le graphe de voisinages contient des circuits ou qu'il est non connexe. Dans notre code R, nous avons utilisé la fonction *isomap* de la bibliothèque vegan. [3] Les principaux hyperparamètres que nous avons dû régler dans cette fonction étaient le nombre de dimensions et le nombre de voisins.

2.1.3 T-distributed stochastic neighbor embedding

L'algortihme t-SNE est basé sur le principe de Stochastic Neighbor Embedding, le principe de base est de remplacer la notion de distance par une probabilité, la probabilité d'être proche. Pour cela nous définissons une distribution gaussienne centrée sur un point x_i et de variance préalablement paramétrée, nous calculons alors la densité des autres points x_j ce qui nous donne, une fois normalisée, la probabilité conditionnelle p_{ij} d'être proche. Pour recréer un espace de plus faible dimension qui conserve ces probabilités, nous allons définir une deuxième densité q_{ij}: une distribution qui modélise la similarité des points y_i et y_j dans l'espace de basse dimension mais cette fois ci basé sur une distribution de student dont la fonction de densité est donnée par :

$$f(t) = \frac{1}{\sqrt{\nu}B(\frac{1}{2}, \frac{\nu}{2})}(1 + \frac{t^2}{\nu})^{-\frac{\nu+1}{2}} \tag{2}$$

Avec ν le degré de liberté, Γ la fonction Gamma et B la fonction Beta.

L'objectif principal de cet algorithme est d'approximer la distribution q_{ij} à la distribution de probabilité p_{ij} en minimisant la divergence de Kullback-Leibler entre ces probabilités conjointes dont nous rappelons l'équation ci-dessous :

$$KL(P \,||\, Q) = \sum_{i \neq j} p_{ij} log \frac{p_{ij}}{q_{ij}} \tag{3}$$

Pour cela l'algorithme utilise une descente de gradient. Pour paramétrer cette méthode il suffit d'ajuster la perplexité qui est l'écart-type de la distribution normal centrée sur les points dans l'espace d'origine et le nombre d'itération de la descente de gradient. En pratique cette technique donne d'excellents résultats car elle transforme l'espace de manière à regrouper les objets similaires et à séparer les données dissemblables ce qui peut-être très utile pour faire par la suite du clustering.

Nous avons utilisé la fonction *Rtsne* qui implémente le t-SNE dans R. [4]. Dans cette implémentation de fonction par défaut, une ACP initiale est effectuée pour réduire la taille de dimension de l'ensemble de données. De plus, deux hyperparamètres peuvent être réglés pour améliorer la formation de la méthode, le nombre d'itérations et la perplexité.

2.2 Méthode d'estimation de la dimension intrinsèque

Une question centrale dans la réduction de dimension est l'estimation de la dimension intrinsèque. La dimension intrinsèque d'un ensemble de données est une mesure de sa complexité. Les ensembles de données qui peuvent être décrits avec précision à l'aide de quelques paramètres ont une dimension intrinsèque faible. Le choix de ce nombre est un processus clé car une estimation de la dimension intrinsèque plus faible peut perdre des informations importantes, tandis qu'une estimation plus élevée peut laisser trop d'informations redondantes, augmentant ainsi la quantité de calculs et masquant les caractéristiques importantes. Dans notre travail, nous avons utilisé deux techniques qui aident à trouver ce paramètre idéal dans des données à haute dimension : l'analyse en composantes principales et l'estimateur de dimension de corrélation.

2.2.1 Analyse en Composantes Principales (ACP)

L'analyse en composantes principales (ACP) peut être utiliser pour estimer la dimension locale de la variété en trouvant la projection des données en dimension inférieure qui préserve au maximum la variance des données. La variance qui peut se traduire comme la quantité d'information.

Cette méthode peut être utilisée pour réduire la dimension, mais elle peut aussi être utilisée pour estimer le nombre de composantes qui résume suffisamment bien les données. Elle repose sur la décomposition de la matrice des données en valeurs et vecteurs propres. Une fois les valeurs propres triées dans l'ordre décroissant elle nous donne une informations sur la quantités d'informations apportée par chaque composante. Dans notre exemple, nous avons établi que le nombre de composantes ayant plus de 80% de la variance était une bonne valeur de dimension.

Cependant, selon [1] l'ACP est un outil puissant pour découvrir la dimensionnalité des ensembles de données avec une structure linéaire ; elle devient cependant inefficace lorsque les données ont une structure non linéaire. Nous avons donc décidé d'utiliser également une autre méthode d'estimation.

[3] https://www.rdocumentation.org/packages/vegan/versions/2.4-2/topics/isomap

[4] https://www.rdocumentation.org/packages/Rtsne/versions/0.15/topics/Rtsne

Afin d'effectuer l'ACP, nous avons utilisé la fonction *prcomp* de la bibliothèque de stats. [5]

2.2.2 Correlation Dimension Estimator

Cette technique permet de calculer de manière rapide la dimension intrinsèque d'un jeu de donnée. Principalement utilisé pour calculé la dimension d'objet n'ayant pas de dimension entière tel que les fractales elle peut également être utile pour trouver la dimension intrinsèque d'un jeu de données plus standard. Cette méthode repose sur l'intégrale de dimension $C(\epsilon) = \lim_{N\to\infty} \frac{g}{N^2}$ (où g est le nombre points qui ont une distance avec un autre point inférieur à ϵ et N est le nombre de point total. L'idée est de calculé $C(\epsilon)$ pour différentes valeurs de ϵ. Enfin en traçant log(C(ϵ)) en fonction de log(ϵ) nous pouvons regarder la valeur de la pente former pour déterminer la dimension intrinsèque.

Nous avons utilisé l'implémentation de l'estimateur de corrélation de la bibliothèque Rdimtools avec la fonction *est.correlation*.[6]

2.3 Métriques d'évaluation

L'évaluation d'une solution est essentielle pour comparer les différentes techniques de réduction de la dimensionnalité. Chaque méthodes présentent des avantages spécifiques et des cas d'utilisation optimale. Nous présentons ici deux métriques clés utilisées dans la littérature pour évaluer si une réduction de dimension est satisfaisante.

2.3.1 Trustworthiness & Continuity

Ces métriques ont été proposées par Venna et Kaski [2] et sont axées sur l'évaluation des performances de préservation du voisinage et de maintien de la structure globale.

Considérant un jeu de données composé de $x_i, 1 \leq i \leq n$ points dans l'espace original à haute dimension, et l'ensemble correspondant composé de points $y_i, 1 \leq i \leq n$ dans l'espace de projection. L'ensemble des points qui ne sont pas dans le k-voisinage de x_i alors que leurs points correspondants sont dans le k-voisinage de y_i est noté $U^k_{x_i}$. Au contraire, $V^k_{y_i}$ est le sous-ensemble qui n'est pas dans le k-voisinage de y_i. Soit $r(x_i, x_j)$ la distance de rang du point x_j par rapport à x_i et k le nombre choisi de voisins. La mesure de Trustworthiness et Continuity, sont définies respectivement comme :

$$T(k) = 1 - \frac{2}{nk(2n-3k-1)} \sum_{i=1}^{n} \sum_{x_j \epsilon U^k_{x_i}} (r(x_i,x_j) - k) \quad (4)$$

$$C(k) = 1 - \frac{2}{nk(2n-3k-1)} \sum_{i=1}^{n} \sum_{x_j \epsilon V^k_{x_i}} (\hat{r}(x_i,x_j) - k) \quad (5)$$

La projection est fiable ou continue si la mesure de qualité correspondante est proche de 1. Selon [3], mesurer si la projection est fiable consiste d'abord à sélectionner les k points les plus proches de y_i dans l'espace de projection puis, les points correspondants dans l'espace d'origine sont identifiés. S'ils sont tous dans le k-voisinage de x_i, alors la projection est entièrement fiable.

Les implémentations de ces métriques dans R sont tirée de la bibliothèque *csoneson/dreval*, faite pour évaluer les représentations de dimension réduite. [7]

3 Données

Pour comparer les différentes méthodes nous avons simulée 4 jeux de données ayant chacun ces spécificités. De manière générale nous les avons choisis dans le but de mettre en évidence les différents points forts et points faibles des méthodes présentées précédemment.

3.1 Swissroll

Le premier jeux de données est assez répandu lorsqu'il s'agit de l'étude des variétés. Il s'agit du Swisroll. Représenté comme une feuille de papier que nous aurions enroulée sur elle-même, voir Figure 1.

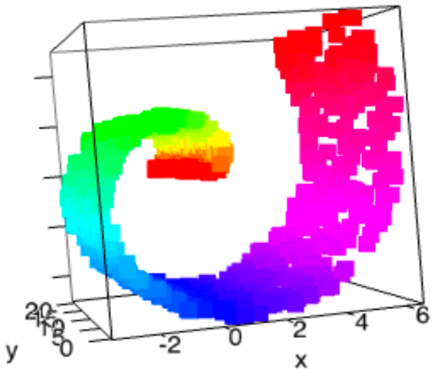

Figure 1. Swissroll data en dimension 3D

Cette structure enroulée pose de réel problème aux méthodes linéaires. L'objectif ici est que l'algorithme « déroule » la structure. Pour simuler ces données nous avons implémenter une fonction qui reprenait l'équation implémentée dans la librairie Scikit Learn pour sa fonction *sklearn.datasets.make_swiss_roll* et nous avons construit ce volume avec mille points.

[5] https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/prcomp

[6] https://www.rdocumentation.org/packages/Rdimtools/versions/1.0.0/topics/est.correlation

[7] https://rdrr.io/github/csoneson/dreval/

3.2 Anneau de Borromée

Le second jeux de données est une structure nommé les Anneaux de Borommée. Ils s'agit de 3 cercles qui s'entrecroisent. Voir Figure 2. Cette entrelacement des données peut être difficile à gérer. Nous l'avons construit avec mille points, avec environ 300 points pour chaque anneaux. Nous espérons que la méthode de dimensions comprenne qu'il s'agisse de trois sous-structures. Ici l'anneau rouge entoure le anneau vert qui lui-même entoure le anneau noir. Nous avons implémenté l'équation paramétrique de ces anneaux.

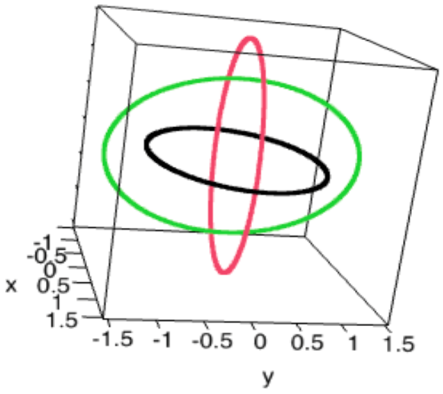

Figure 2. Anneaux de Borromées data en dimension 3D

3.3 Sphère Unitaire

Le troisième jeu de données est un jeu de donnée simuler selon la proposition de P. Desmartines dans sa thèse de 1994. L'objectif de cette méthode est construire une variété dans un espace de dimension faible, dans notre étude nous avons choisi une sphère unitaire de dimension 3. Cette structure va être plongé dans un espace de dimension supérieur (dimension 10).

Nous espérons pour ce jeu de donnée que la méthode de réduction de dimensions parviendra à retrouver la dimension original de la variété dans notre cas la sphère de dimension 3.

3.4 Données réelles: Wine

Ce jeu de données est souvent utilisé pour des tâches de classification. Il consiste en une analyse chimique de vins cultivés dans une région d'Italie mais issus de trois viticulteurs différents. L'analyse a déterminé les quantités de 13 constituants trouvés dans chacun des trois types de vins. Nous avons chargé ces données en utilisant la bibliothèque *bootcluster* de R, qui fait référence aux données du site https://archive.ics.uci.edu/ml/datasets/wine. Nous avons un dataframe avec 178 échantillons et 14 variables, la première colonne est la classe qui définit les types de vins

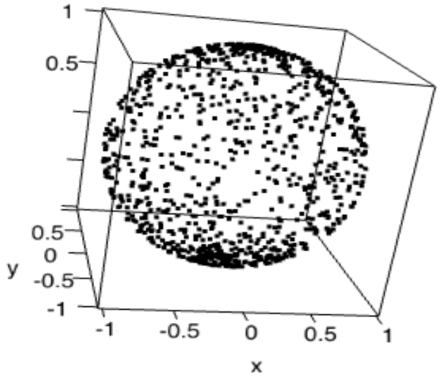

Figure 3. Sphère data en dimension 3D

et les 13 autres variables sont des propriétés du vin. Les 13 caractéristiques sont : l'alcool, l'acide malique, les cendres, l'alcalinité des cendres, le magnésium, les phénols totaux, les flavanoïdes, les phénols non flavanoïdes, les proanthocyanines, l'intensité de la couleur, la teinte, les OD280/OD315 des vins dilués et la proline.

4 Expériences numériques sur les données artificiels

4.1 Swissroll from 3D to 2D

Dans cette première expérience, nous avons travaillé avec le jeu de données Swissroll, un manifold complexe connu. Le but de cette expérience était de trouver une technique de réduction qui puisse déplier le rouleau de manière à ce qu'il soit possible de visualiser les différentes couleurs sans chevauchement. Les hyperparamètres de t-SNE utilisés étaient la perplexité par défaut égale à 30 et l'itération maximale égale à 1000. Dans le cas d'Isomap, nous avons fixé à 10 le nombre de voisin utilisés pour construire le graphe de voisinage. Le modèle MDS n'a pas de paramètres à régler autres que la taille des dimensions.

La figure 4 présente les résultats obtenu après l'utilisation des techniques t-SNE, MDS et Isomap pour transformer les données 3D en une représentation bidimensionnelle.

Compte tenu de ces résultats, nous avons analysé que la méthode Isomap était l'algorithme qui a réussi le mieux à atteindre notre objectif de dérouler les données de manière à mieux visualiser chaque couche colorée. Après cela, nous avons décidé d'ajuster le paramètre du nombre de voisins d'Isomap pour obtenir le meilleur résultat possible. Nous avons effectué une recherche exhaustive de valeurs allant de 6 à 16 en augmentant par puissance de 2 afin de trouver le résultat optimal. Les valeurs inférieures à 6 fragmentaient les données et Isomap n'était pas en mesure de construire le graphe. Les résultats sont présentés dans la figure 5. Nous avons choisi de travailler avec K=12, parce que c'était le plus petit nombre de voisins qui montrait

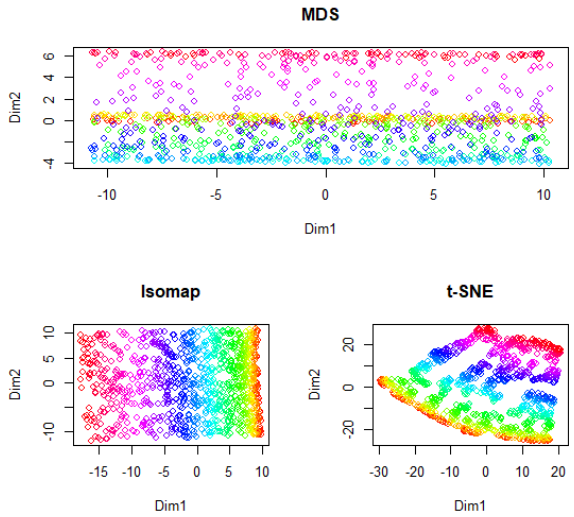

Figure 4. Application des techniques de réduction de la dimensionnalité des données Swissroll 3-D à 2-D

déjà une séparabilité compacte et que pour K plus grand, le résultat était fondamentalement le même.

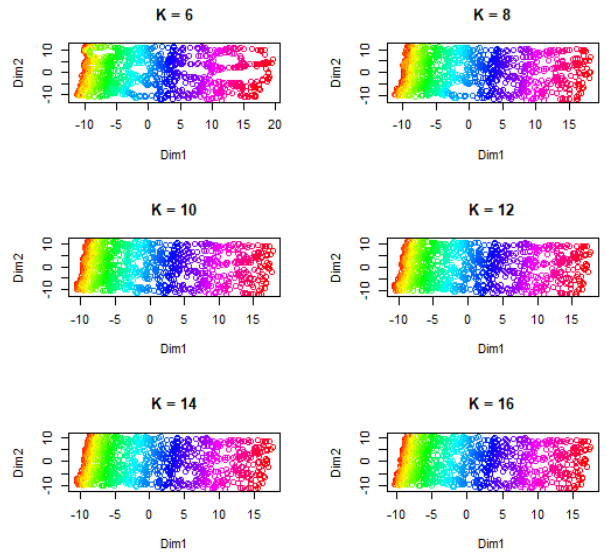

Figure 5. Expérimentation de différents nombres de voisins pour la méthode Isomap

En résumé, nous avons montré que pour un jeu de données similaire à Swissroll, l'Isomap avec un nombre de voisins d'environ 12 pouvait le représenter avec succès dans un espace bidimensionnel.

4.2 Sphere en haute dimension: espace à 10 dimensions

Dans cette expérience, nous avons utilisé les données de la sphère immergée dans un espace à 10 dimensions. Comme ces données étaient dans une dimension élevée, nous avons utilisé les deux techniques d'estimation de la dimension intrinsèque expliquées précédemment (ACP et Correlation Estimation) pour estimer la meilleure dimension afin de réduire la dimension des données.

Avec l'ACP, nous avons constaté que plus de 80% de l'information est exprimé à travers les trois première composantes. Dans le graphique de la figure 6, nous pouvons voir la proportion de la variance en fonction du niveau de la composante. D'autre part, l'analyse de l'estimateur de corrélation a également donné une valeur de 3.

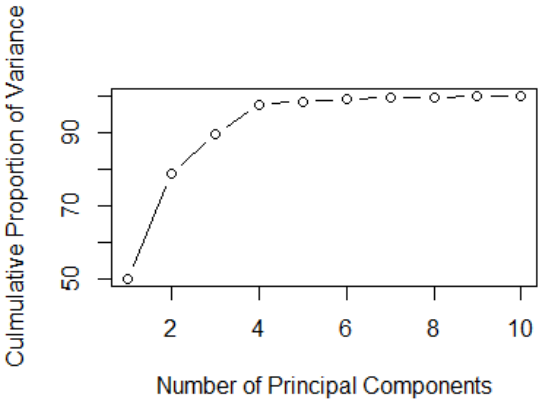

Figure 6. ACP : Variance cumulée en fonction du nombre de composantes

Ainsi, nous avons effectué la réduction de dimension de ces données en passant de 10-D à 3D. Notre objectif principal était de trouver une technique capable d'effectuer cette réduction tout en conservant la forme particulière de la sphère.

Nous avons donc appliqué les techniques t-SNE , MDS et Isomap pour effectuer cette réduction. Pour cette tâche spécifique, nous avons choisi par expérimentation le nombre de voisins k=10 pour isomap et les valeurs par défaut des hyperparamètres pour t-SNE. Dans la figure 7, nous voyons les résultats de chaque méthode.

Comme nous pouvons le constater, la technique Isomap est celle qui a réussi à transformer les données dans un format qui ressemble plus à une sphère. t-SNE a montré le plus mauvais résultat pour cette expérience car elle a essayé de regrouper le voisinage en clusters et non en une surface continue.

En conclusion, Isomap est la technique qui a réussi à mieux maintenir l'aspect des données dans la dimension

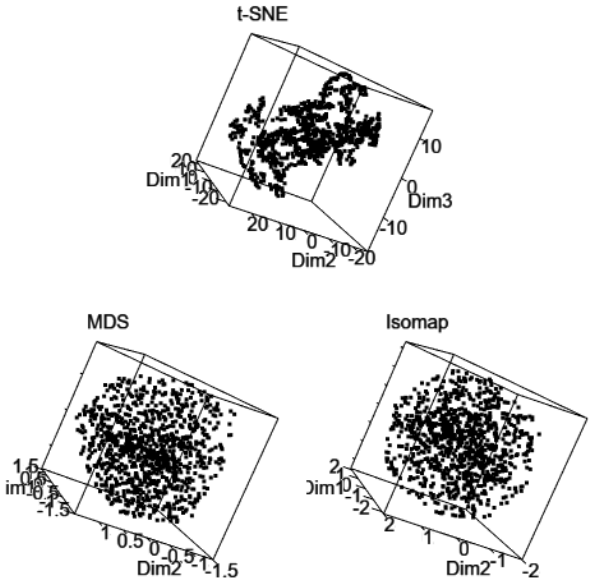

Figure 7. Résultats de la réduction d'une sphère à 10 dimensions à l'espace 3-D

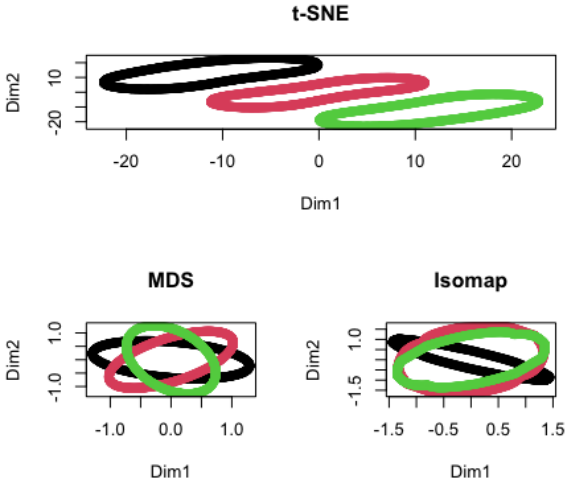

Figure 8. Application des techniques de réduction de dimension des données 3D des Anneaux Borromée aux données 2D

original (voir Figure 3) avant qu'elle ait été plongé dans un espace de plus grande dimension.

4.3 Anneau de Borromée

Passons à présent aux résultats obtenus pour les Anneaux de Borromée. Nous le rappelons il s'agit de trois cercles entrelacés qui sont représentés en 3 dimensions. L'objectif va être de les représenter au mieux dans un espace en 2 dimension. Nous attendons des techniques qu'elles distinguent bien les trois cercles afin de séparer trois groupes au sein des données et si possible que ces derniers conservent leur forme de cercle. Voyons dans un premier temps les résultats obtenus (voir Figure 8) pour les trois méthodes calibrées pour avoir la meilleure représentation.

Pour la calibrations nous avons utilisé une perplexité de 100 pour t-SNE avec 500 itérations, et pour la méthode Isomap k égale à 200.

Nous voyons très clairement que seul t-SNE à réussi à séparer les trois anneaux. Cependant bien que le résultat est aplati, les formes recréées ne ressemblent pas à des cercles mais plus à des ellipses. Nous constatons ici la faculté de t-SNE à retrouver les différentes sous variétés qui constitues le jeu de données.

Pour apporter quelques précisions sur la calibration de la méthode t-SNE nous avons constaté qu'une perplexité de 50 et 100 itérations l'algorithme parvenait déjà à séparer les cercles mais cela n'était pas stable et en relançant plusieurs fois nous tombions sur des résultats très différents. (Voir Figure 9) De plus augmenter la perplexité et le nombre d'itération n'était pas beaucoup plus coûteux en temps. Voilà les raisons qui nous ont poussé à faire ce calibrage.

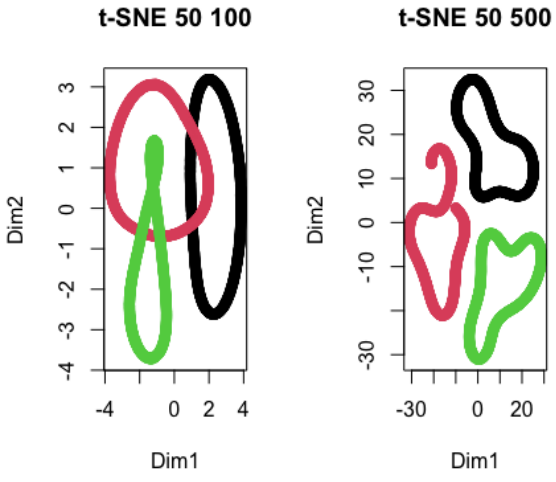

Figure 9. Application de t-SNE avec une perplexité fixée à 50 dans les données 3-D des anneaux Borromeo à 2-D

5 Comparaison des méthodes

Nous avons choisi quelques approches pour comparer les méthodes : le temps de calcul, la Trustworthiness Continuity et la visualisation. Dans [4], les auteurs ont utilisé le temps de calcul et la visualisation comme moyen de comparer les méthodes de réduction de dimensionnalité linéaires et non linéaires. Cela a permis de voir quelle approche était la plus coûteuse en temps de calcul et celle qui fournissait une visualisation la plus informative. De plus, le travail de [5] a utilisé Trustworthiness Continuity comme forme d'évaluation de la performance des algorithmes de réduction de la dimension.

Nous avons comparé le temps de calcul en utilisant la bibliothèque microbenchmarks. Nous avons réduit chaque donnée artificielle à un espace bidimensionnel trois fois pour chaque méthode et avons vu combien de temps il fallait pour obtenir nos résultats. Nous choisissons ici les mêmes valeurs de paramètres que celles utilisées dans la section précédente pour chaque ensemble de données. Dans le tableau 1 nous avons le temps de calcul moyen des 3 exécutions pour chacune de nos trois méthodes en secondes.

Table 1. Temps de calcul des méthodes de réduction de la dimensionnalité

Méthode	Temps moyen (sec)
Swissroll à 3 dimensions	
MDS	33.504
Isomap	18.284
t-SNE	6.092
Sphère à 10 dimensions	
MDS	28.907
Isomap	13.381
t-SNE	3.577
Anneau à 3 dimensions	
MDS	501.683387
Isomap	211.685454
t-SNE	6.618515

En analysant ces résultats, nous pouvons voir que t-SNE est beaucoup plus rapide que les autres méthodes et qu'elle est capable de réduire un jeu de donnée en 10 dimensions à une variété de 2 dimensions quatre fois plus rapidement que les autres méthodes. De plus, nous remarquons que pour les manifolds très complexes comme les anneaux, MDS et Isomap prennent beaucoup de temps pour terminer la réduction.

Enfin, nous avons utilisé Trustworthiness Continuity pour voir quelle méthode préserve le mieux le voisinage local dans un espace à faible dimension. Dans le tableau 2 nous pouvons voir ces résultats et les comparer pour chaque jeu de données.

De cette façon, nous voyons que toutes les méthodes ont réussi à créer de bonnes réductions des données mais certaines ont préservé davantage le voisinage, notamment

Table 2. Comparaison des résultats des méthodes pour tous les ensembles de données en fonction de la fiabilité et de la continuité.

Méthode	Trustworthiness	Continuity
Swissroll		
MDS	0.9690	0.9986
Isomap	0.9998	0.9998
t-SNE	0.9998	0.9996
Sphère		
MDS	0.9497	0.9955
Isomap	0.9479	0.9957
t-SNE	0.9992	0.9952
Anneau		
MDS	0.9965	0.9999
Isomap	0.9944	0.9999
t-SNE	0.9999	0.9996

Isomap et t-SNE pour Swissroll, t-SNE pour Sphere et les trois méthodes pour les Anneaux.

Comme le montre la section précédente, nous avons analysé l'aspect visuel après la réduction dimensionnelle et, en fonction de chaque objectif spécifique, Isomap a produit un meilleur résultat pour les données Sphere et Swissroll et c'est t-SNE qui a produit la meilleure représentation pour les Anneaux de Borromée.

6 Application sur données réelles

Nous avons utilisé les données réelles de Wine pour appliquer nos solutions. Ce jeu de données a une structure à 13 dimensions avec 3 classes distinctes. Dans un premier temps, nous avons utilisé les techniques d'estimation des dimensions intrinsèques, afin de trouver la réduction appropriée pour ces données pour ne pas perdre d'informations pertinentes. Pour cela, l'ACP a donné une dimension égale à 1 et l'estimateur de corrélation égale à 2. Comme nous l'avons mentionné précédemment, l'ACP pourrait ne pas être en mesure de détecter la meilleure dimension intrinsèque sur des données aux structures non linéaires, nous avons donc utilisé l'estimation de la dimension de corrélation. Ceci dit, nous avons appliqué les trois techniques de réduction de dimension MDS, Isomap et t-SNE pour visualiser cet ensemble de données à 14 dimensions en 2D.

Dans t-SNE nous avons choisi les deux paramètres principaux en faisant des expériences avec une gamme de valeurs et nous avons choisi les résultats qui séparent le mieux les classes. Nous avons expérimenté la perplexité dans une plage de 10 à 20 et le nombre d'itérations de 20 à 1500. Dans la figure 10 nous avons un exemple d'expérimentation et nous pouvons voir l'impact que l'augmentation du nombre d'itérations a dans le résultat, notamment avec 20 itérations les données de classe étaient mélangées et à partir de 500 nous pouvons voir une que les données sont bien séparées.

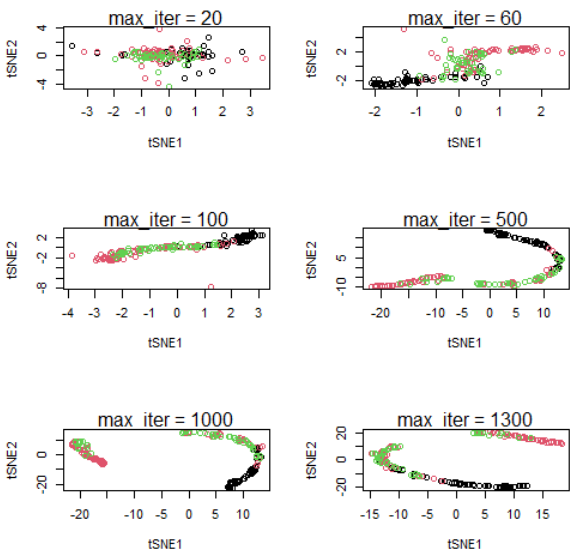

Figure 10. Expérimentation de différentes valeurs du nombre d'itérations pour l'application de t-SNE dans l'ensemble de données Wine.

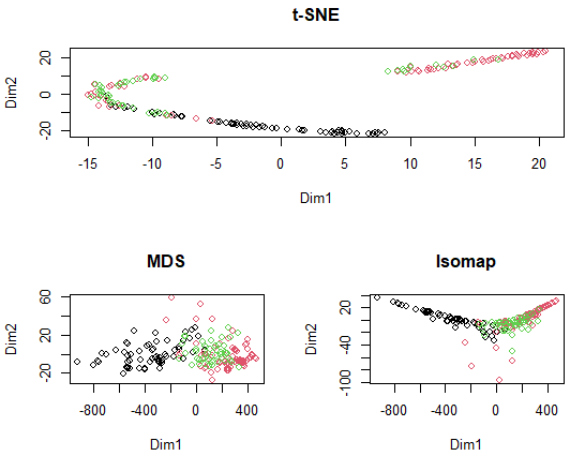

Figure 11. Résultats visuels de l'application de la réduction dimensionnelle sur le jeu de données "Wine" avec trois techniques

Enfin, nous avons choisi une perplexité de 17 et un nombre maximal d'itérations de 1000, car les résultats étaient satisfaisants et il n'était pas nécessaire d'ajouter plus d'itérations et de rendre le traitement plus lent. Dans le cas d'Isomap, nous avons choisi d'avoir un nombre de voisins égal à 10. La figure 11 montre les résultats de la réduction de la dimensionnalité par chaque méthode. Nous pouvons voir que le MDS n'a pas été capable de séparer la classe cible et que le t-SNE est la méthode qui a mieux regroupé les classes.

De plus, nous avons vérifié le temps de calcul en exécutant 20 fois chaque solution et nous avons calculé la Trustworthiness & Continuity pour comparer ces solutions. Au vu des résultats du tableau 3, nous remarquons que, comme l'ensemble de données ne comporte que 178 échantillons, le calcul de cette réduction est rapide pour toutes les méthodes, t-SNE étant la plus longue en raison du grand nombre d'itérations que nous avons fixé. De plus, en analysant les mesures d'évaluation, il semble que nous ayons effectué une réduction de la dimension d'un ensemble de données à 14 dimensions à une représentation 2D avec une performance élevée pour toutes les méthodes.

Dans ce cas précis, nous concluons que t-SNE était le mieux adapté car il était encore une fois le plus rapide et parvenait à séparer les classes pour une bonne interprétation et visualisation de l'ensemble de données dans un espace de faible dimension.

Table 3. Comparaison des mesures de temps et d'évaluation dans le jeu de données Wine

Méthode	Temps (μs)	Trustworthiness	Continuity
MDS	101.8527	0.9994	0.9996
Isomap	64.282	0.9963	0.9977
t-SNE	168.7512	0.9984	0.9986

7 Conclusion

Dans ce travail nous aurons expérimenter différentes méthodes de réduction de dimension appliquées à des données artificielles et réel. Nous avons dans un premier temps rappelé quelques principes théoriques des méthodes utilisées. Nous avons conçu trois jeux de données avec différentes caractéristiques. Nous avons enfin appliquer les méthodes choisies aux différents jeux de données puis avons proposer quatre critères de comparaison pour comparer les résultats obtenus. Toutes les méthodes présentées repose sur un critère de similarités des points qui doit être reproduit dans un nouvel espace. Nous avons vu que suivant la méthodes utilisée les résultats peuvent être très différents et que suivant les contextes certaines méthodes étaient plus appropriées. Enfin, comparer ces méthodes est relativement facile lorsque le résultat peut être visualiser car nous pouvons déterminer cela de manière graphique mais il est plus difficile de formaliser cette réussite. Pour conclure nous avons proposé une vue d'ensemble du processus de réduction de dimension grâce à cela nous comprenons mieux les différents enjeux de ces méthodes.

References

[1] Mingyu Fan, Nannan Gu, Hong Qiao, and Bo Zhang. Intrinsic dimension estimation of data by principal component analysis. *ArXiv*, abs/1002.2050, 2010.

[2] Jarkko Venna and Samuel Kaski. Neighborhood preservation in nonlinear projection methods: An experimental study. In *ICANN*, 2001.

[3] Victor Onclinx, Vincent Wertz, and Michel Verleysen. Nonlinear data projection on non-euclidean manifolds with controlled trade-off between trustworthiness and continuity. *Neurocomputing*, 72:1444–1454, 2009.

[4] Anna Konstorum, Nathan Jekel, Emily Vidal, and Reinhard C. Laubenbacher. Comparative analysis of linear and nonlinear dimension reduction techniques on mass cytometry data. *bioRxiv*, 2018.

[5] Spyridon Stasis, Ryan Stables, and Jason Hockman. Semantically controlled adaptive equalisation in reduced dimensionality parameter space. *Applied Sciences*, 6:116, 2016.